to step data y n.
pin 4 (RTS) if terminal
pin 5 (CTS) if processor

DTC MICRO FILE COMMANDS

OPTION PARAMETERS IN PARENTHESIS - (DO NOT TYPE BRACKETS THEMSELVES)

NAME	EXAMPLE	FUNCTION
AB	AB(N) *(Abort Background)*	
BK COPY	CO FNAME FT DN FNAME2 ((FT (DN2)) *(PN)*	Copies file 'FNAME' type 'FT' on DN to DN2 and names it 'FNAME2'. If DN2 is not specified new file goes on DN.
CP (copy & remove parity bit on 'T' files)		
CWL	CW	Changes character, word & line delete characters to $A^C B^C C^C$
	CWL = $A^C B^C C^C$	(carriage return after = will revert to BS,DEL & X^C).
BK DDUMP	DD HEXFROM (HEXTO) (DN) *(PN)*	Dumps disk sectors (hex representation). First 2 digits = track, second = sector.
	(S-2) *FT*	
EDIT	ED FNAME *(DN)*	Invokes Micro File Text Editor-- See editor commands.
ERASE	ER FNAME FT (DN)	Erases file 'FNAME' type 'FT'.
EXEC	EX FNAME *FT* (DN)	Executes series of system commands stored on 'FNAME' as above only prints commands before they are executed.
	EXP FNAME (DN)	
BK FDUMP	FD FNAME FT (DN) *(PN)*	Dumps hex form of file 'FNAME'.
FILES	FI (DN) *(PN)*	Lists file and file types
BK FL		
FORMAT	FO DN	Writes all sectors of disk (format enable switch must be on)
BK FS	(LABEL = XXX...	(Label 'XXX...' was previously written on disk, Do you wish to reformat? Answer 'Y' if yes, 'N' if no).
	REFORMAT?)	
	LABEL = NEW LABEL	Enter name of label.
GO	GO	Program control transferred to just loaded program.
	GO HEXLOC	Transfers program control to location 'HEXLOC'.
	GO PARM1 PARM2...	Program executes and uses PARM1 PARM2 etc. as its parameters.
HLLOAD	HL (HEXBIAS) *(PT)*	Retrieves a program from a HOST system, loading it into RAM.
HTLOAD	HT (HEXBIAS)	Loads 'HEX' paper tape into RAM. Stores at its origin address +HEXBIAS (if any). See SAVE command).

FS DN T N ← # OF ENTRIES/LINE
       L

NAME	EXAMPLE	FUNCTION
LABEL	LA (DN) LABEL = XXX... NEW LABEL =	Reads label of disk. If new label not required type CR.
LOAD	LO FNAME (DN)	'FNAME' loaded into RAM but not executed.
BK MDUMP *DU HEXFROM HEXTO (L)*	MD HEXFROM (HEXTO) *(PN)*	Dumps memory starting at HEXFROM (16 bytes/line).
PATCH	PA HEXFROM HEXBYTE...	Patches data into RAM starting at HEXFROM.
BK PRINT *X PT N* * PL N*	*FT* PR FNAME~(DN) *PN*	File FNAME will be printed at terminal. *X = X-ON, Y-OFF*
RCEIVE *P RB(P) - receives 'binary' info (does not exclude nulls)* *RX* *RA YXX PN*	RC FNAME T (DN) ~~(PRINT)~~*(PN)* *P* *DN*	Retrieves a text file from a HOST system, storing it as a local file. P option prints as it receives.
RNAME	RN FNAME FT, FNAME2 (FT2) ~~(DN)~~	Renames 'FNAME' to FNAME2'
RUN	RU FNAME (PARM1 PARM2...)	'FNAME' loaded into RAM and executed against parameters PARM1 PARM2...etc.
SAVE	SA FNAME HEXFROM HEXTO HEXLOAD (DN)	Causes file 'FNAME' to be created. (Type P).
BK SEND	*FT* SE FNAME ~~FT~~ (DN) *PN* *SE(P)(L)*	Transmits the file FNAME T to the HOST system.
SIZE	SI FNAME FT (DN)	Prints size of file in sectors and bytes if FT = P. In sectors and characters if FT = T.
BK SND	*FT* SN FNAME ~~R~~ (DN) *PN* *SN(P)(L)*	Same as SEND except XOFF and XON protocol is observed.
SPACE	SP (DN)	Shows amount of free space in sectors and percent.
SW (SWITCH TO NEW SYSTEM) *SX (P)(L) FNAME (DN)*		
BK SZ	SZ (DN) *(PN)*	Lists files, types and sizes.

RUN FMT OFN DN NFN DN   *RUN 22600 FN DN HEX-OFFSET HEX-SIZE [V]*
  FUZZ

RUN PPHEX FN FT DN (L)

RUN PPLOG FN FT DN √REL-ADDR CHIP-SIZE (L)
                              (IN HEX)

[AL] [0]
[AS] [1] FNAME [DN [HEXBIAS]]   ASI = NO LISTING
*   [2]*
*   [3]*

RUN SWAT (DISK-ADDR) DN   *R... {DISK} SADDR EADDR (L)   DIS8 A800   760601*
                                                             DIS84: 2800

TEXT EDITOR COMMANDS -

Command:	Syntax:	Function:
AGAIN	A	Executes the preceding command.
	Ann	
	A*	
BOTTOM	B	Positions to the last line of the file.
	*	
DELETE	D	Deletes line(s).
	Dnn	
	D*	
EXIT	E	Exits from the Editor.
GET	G FNAME(DN) (LFROM(NO.))	Gets lines from the file FNAME T and inserts them into the file being edited.
INSERT	I TEXT	Inserts text following the current
	I	line, or begins "block" insert mode.
NEXT	N	Positions downward in the file
	Nnn	
PRINT	P	Prints line(s).
	Pnn	
	P*	
PRINT-NUMBERED	PN	Prints line(s) with prefixed numbers.
	PNnn	
	PN*	
PRINT-CONTROLLED	PX	Prints line(s) under control XOFF and
	PXnn	XON characters.
	PX*	
UP	U	Positions upward in the file.
	Unn	
REPLACE	R TEXT	Replaces line(s) with a single line of
	Rnn TEXT	text.
	R* TEXT	
	R	Replace line(s) with multiple lines of
	Rnn	text.
	R*	
TOP	T	Positions to the top of the file.
Line nn	nn	Positions to line nn of the file.
String-Search	'string'	Positions downward in the file to the
	"string"	line containing 'string'.
String-Replace	'string1='string2'	Replaces occurrence(s) of 'string1'
	"string1="string2"	with 'string2'.

AX
*: c/R, AT: c/R, c/F
⌈AX⌉ HEXBIAS FN DN ⌈FN DN ···⌉
⌊AE⌋

HF HEXBIAS FN DN ⌈FN DN ···⌉ loads 8080 object
programs (FT OF O)

Ann ... DTC LDX cx ⌈PD CEN⌉

MODIFY COMMAND

M   Modify (nn) lines
Mnn

in the descriptions below:
    n - optional repeat count
       - any character

Command	Syntax	Description
again	A	restart line editing, cancel all changes
back	n(Backspace*)	moves pointer left by character
change	NCc.....c	change characters in line
delete	nD	delete characters
end	E	complete editing, but don't print remainder
extend	X	extend line
finish	(CR)	complete editing line, print
half-extend	H	replace remainder of line
insert	I	insert text until ATTN or CR
justify	J	break line into two lines
kill	nKc	skip and delete to character
line	L	print remainder of line
next	n(Space)	moves pointer right
print	P	print line and reposition to current character
quit	Q	terminate modification and exit to editor command level
replace	nR	same as nDI
skip	nSc	skip right to character
word	n(DEL)**	moves pointer left by words

---

*The system "character delete" should be typed. The default is (Backspace)

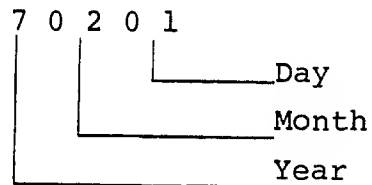** The system "word delete" should be typed. The default is (DEL)

.

MICRO FILE NEWS - 70201

1. Sequencing of Micro File News Sheets

   In future, all Micro File News sheets will have a
   5-digit sequence number that shows the date in the
   following format:

```
7 0 2 0 1
|   |   |_____Day
|   |_____Month
|_____Year
```

2. 1 or 2 Stop Bits Per Character

   Normally at 110 baud, 2 stop bits are added to each
   character. At all other speeds only 1 stop bit is added.
   If the back panel switch is set at 110 baud and the port
   speed is changed by RA, then each character will have
   2 stop bits regardless of speed. When the back panel
   switch is set at a speed other than 110 and the port is
   changed by RA to 110, then each character will have
   only 1 stop bit.

3. Editor Tip

   If you wish to display (or print) every occurrence of
   a string of characters in a file, use the following
   technique:

   -T

   -"string"

   -A*

MICRO FILE NEWS - 70203

DOCUMENT PROCESSOR

Version 1 of the Document Processor is now in initial release.
It allows preparation of 'camera ready' manuals and documents.
Some of the features are:

VERTICAL LAYOUT

```
PL - Page length
TM - Top margin
HM - Heading margin
FM - Footing margin
BM - Bottom margin
```

HORIZONTAL LAYOUT

```
LL - Line length
MR - Margin size (left side)
IN - Indent crown line of paragraph
OF - Offset vest lines of paragraph
RI - Right indent
```

HEADINGS

```
HE - Define text for all headings
HL - Define text for left heading only
HR - Define text for right heading only
```

FOOTINGS

```
FT - Define text for all footings
FL - Define text for left footing only
FR - Define text for right footing only
```

PAGE CONTROL

```
PN - Page number control
RH - Force right-hand page format
```

TEXT FORMATTING

```
BR - Break (force end of) paragraph
TR - Transparent text format
NF - No format (with margin control)
CE - Center text
CO - Concatenate text to fill lines
JU - Justify text to right boundary
```

SPACING

    SS - Single space text
    HS - One-and-a-half space text
    DS - Double space text
    PA - Go to new page
    CP - Conditionally page
    SP - Space insert
    US - Unconditionally space
    CS - Contiguous space insert

CONTROL

    IC - Indicate change (change bars)
    TB - Define tab stops
    CM - Comment
    CC - Change control character

Cost of this package is $500 which includes update to
Version 2 that is planned to have the following features:

    Double columns on a page
    Heading styles built in
    Table of Contents
    Index
    Footnotes
    Proportional spacing for DTC-382

Update service is $100 per year and is similar to the General
System and BASIC update services.

A user manual is available for $3 if you wish to review the
full capabilities before purchasing.

DEBUG CONSOLE

These units are now ready for shipment.   Enclosed is a brochure
describing the unit.   Cost is $500.

MICRO FILE NEWS - 70321

BASIC MANUAL

An error has been found on Page 28 of the new BASIC Manual.
Under the MON command it is not possible to use RN (rename)
or CO (copy), as stated in the second paragraph. These
system commands use space above 2800_H and thus interferes
with BASIC. Please delete RN and CO in your copy of the
manual.

REFERENCE CARD

There is a typographical error under the Names File Maintenance
description. The command name PR(n) should read PRI(n).
Please make this correction on your reference card(s).

DTC MICRO FILE
BASIC INTERPRETER

STATEMENTS		COMMANDS	FUNCTIONS	
IF...THEN[1]..ELSE	DIM	LIST	RND	LOG
GOSUB	REM	RUN	SQR	EXP
RETURN	RESTORE	CLEAR[5]	SIN	TAN
FOR	PRINT	SCRATCH	ABS	ATN
NEXT	ERASE	DELETE	INT	FRE[7]
READ	SWAP	CONT[6]	SGN	POS
INPUT	TRON	EDIT	COS	
STOP	TROFF	PEEK		
PRINT USING	ON...GOTO	POKE		
END	ON...GOSUB	NULL		
DATA	DEF[4]			
LET[2]	INTEGER DIVISION			
TAB[3]	TYPE			
MOD				

NOTES:

[1] IF...THEN can be followed by a statement. Example: IF A $<$ 5 THEN PRINT B.

[2] LET is optional in variable assignments. Example: A=5 is identical to LET A=5.

[3] TAB(X) within PRINT statement tabs to print column X.

[4] DEF allows for single variable single statement user defined function.

[5] CLEAR deletes all variables.

[6] CONT continues program execution after Control C or STOP.

[7] FRE returns number of free bytes for program or variable storage. With a string argument, FRE returns amount of free string space.

FEATURES

●Multiple statements per line, separated by a colon ":" (132 characters per line).

●"X^C" deletes a whole line, rubout deletes a word, and "backspace" (or N backspaces) deletes last character(s) typed.

●Direct execution of any statement except INPUT.

●Full text error messages

●Control C - interrupts program (prints BREAK IN LINE XX).

●Control O - toggles suppress output switch.

●All results are calculated to at least six decimal digits precision. Exponents may range from 10^{-38} to 10^{37}.

●Maximum line number of 65,535 .

●Multi-dimensioned (up to 255) arrays for both strings and numbers.

●STRINGS

    Maximum length = 255 characters

    String concatenation (A$ + B$)

    LEN   - length of string

    ASC   - returns the equivalent ASCII decimal number of the specified argument.

    CHR&   - Truncates the numeric formula to an integer, interprets the integer as a decimal number, and converts it to its equivalent ASCII character.

    RIGHT$ - Return substrings of specified string formulas; beginning
    LEFT$ - as leftmost character (LEFT$) or ending rightmost (RIGHT$)
    MID$ - or beginning at specified position (MID$) of the string formula, and containing the number of characters specified by the numeric formula.

    STR$ - number converted to a string

    VAL   - string converted to a number

    EDIT   - allows editing within lines of a program

    DISK FEATURES

●KILL   - Deletes file on disk specified

●OPEN   - Opens file (by mode) on disk specified

●CLOSE   - Closes the file(s) given

●INPUT   - From file

●PRINT#   - To file

●LINE INPUT - Reads string onto file

●GET   - Random read of file

●PUT   - Random write of file

●LOAD   - Loads program file into memory from disk

●SAVE   - Saves file on disk

    ETC

MICRO FILE NEWS - 70712

In order for DTC Micro File Users to keep abreast
of developments in software, etc. DTC has offered
an update service for General Systems and Extended
BASIC updates. At this time we would like to
announce that update service is now available for
Document Processor also.

DTC is charging $100 per year for Document Processor
updates. Likewise to the other update services
available, users will be required to initially have
blank disks stored at our main plant so that upon
release of an update we can automatically forward
a disk with those changes on it to the registered
update service user.

Version 1.3 of Document Processor is ready for
release. An Update Service order form is attached
for your convenience.

DTC MICRO FILE
Software Update Service Order Form

TO: DTC
     1190 Dell Avenue
     Campbell, CA 95008

PLEASE SUPPLY (check):

_____ 12 Months General System Updates @ $100

_____ 12 Months Extended Basic Updates @ $100

_____ 12 Months Document Processor Updates @ $100

OTHER:_____

PLEASE NOTE: Should no updates occur during
the 12 month period, either:

A) A refund will be given

B) Services will be extended for another 12
    months.

5 BLANK DISKS:

_____ To be included in this order @ $60

_____ Are enclosed

_____ Will be forwarded shortly

ENCLOSED IS:

_____ Check in the amount of $_____

_____ P.O. (for approved accounts) for $_____
      P.O. number is_____

SHIP TO:

_____   USER CONTACT_____

_____   PHONE NUMBER_____

_____   SIGNED_____

_____   DATE_____

FOR DTC OFFICE USE ONLY   _____

DISK #1_____   DATE_____   _____

DISK #2_____   _____

DISK #3_____   _____   _____

DISK #4_____   _____

DISK #5_____   _____   _____

70712

MICRO FILE NEWS - 770715

Null and Delete Characters in Text Editor

These characters are treated in a special manner. If you
search a file containing null or delete characters using
the string search capability (using ' or "), they will
not be found under normal circumstances. e.g. 'Null' = 'ABC'
is equal to '' = 'ABC' i.e. the null is ignored.

In order to work on these characters, put the Transparent
switch on the front panel on. With this switch on you must
terminate each sequence of terminal input with a return
and a line feed.

To Change the Assembler Output to DCL instead of DTE


```
* LO $AS$ D1 1000
* PA 3715 A9 21
* PA 371D 26 21
* PA 3725 03 08
* PA 3737 0F 08
* PA 378C 26 21
* PA 3794 03 08
* SA $AL$ 3700 4144 2700
```


770715

MICRO FILE NEWS - 770817

GENERAL SYSTEM UPDATE

LETTER WRITER - SYSTEM COMMANDS

I. Letter Writer - Backspace problem

    The Letter Writer experiences line justification
    problems when the backspace key is used to go back
    and underscore a word. The new version allows the
    underscore of words and phrases. A word is under-
    scored by immediately backspacing and underscoring
    the word. A phrase is underscored by typing a
    printing character between each word of the phrase
    e.g. "DTC_MICRO_FILE", and then backspacing to
    underscore. Figure 1 shows the justification problem
    and operation of the corrected version.

II. System Commands - SI, SZ Binary File Calculation

    The SIze and SZ commands for listing the size of
    a file and all files respectively have character
    count errors when binary files are encountered. The
    sector counts are correct in all cases.

    For example: *SI TRADE B D1

                    TRADE B=0053 SECTORS...004291 CHARS

    Corrected: *SI TRADE B D1

                    TRADE B=0053 SECTORS...006530 CHARS

    And also: *SZ D1
                    LIFE2 T = 0017 SECTORS...002065 CHARS
                    23MCH T = 0007 SECTORS...000842 CHARS
                    WMPUS T = 0046 SECTORS...005658 CHARS
                    TRADE B = 0053 SECTORS...004291 CHARS
                    TRADS B = 0110 SECTORS...007693 CHARS
                    STTRK B = 0120 SECTORS...015010 CHARS

```
Corrected:       *SZ D1
                 LIFE2 T = 0017 SECTORS...002065 CHARS
                 23MCH T = 0007 SECTORS...000842 CHARS
                 WMPUS T = 0046 SECTORS...005658 CHARS
                 TRADE B = 0053 SECTORS...006530 CHARS
                 TRADS B = 0110 SECTORS...013627 CHARS
                 STTRK B = 0120 SECTORS...014984 CHARS
```

Subscribers to the Update Service will be receiving
their updated disks shortly.

FIGURE 1

This is a test letter on the DTC MICRO FILE to illustrate the
backspace problem. The backspace is not
counted properly in Version 1 of Letter Writer. This causes
incorrect line justification.

An inappropriate way to use underscore is to go back and
underline the entire phrase after it was typed. The DTC
MICRO FILE   will not work in all
cases as this example shows.

The correct way is DTC MICRO FILE,
as you see no justification problems. In order to have a
phrase underlined with contiguous underscores it is
neccessary to type and underscore between each word of the
phrase instead of a space. The
DTC MICRO FILE is an example of
the technique.

Corrected Version:

This is a test letter on the DTC MICRO FILE to illustrate
the backspace problem. The backspace is not counted properly
in Version 1 of Letter Writer. This causes incorrect line
justification.

An inappropriate way to use underscore is to go back and
underline the entire phrase after it was typed. The DTC
MICRO FILE   will not work in all cases as this example shows.

The correct way is DTC MICRO FILE, as you see no
justification problems. In order to have a phrase underlined
with contiguous underscores it is neccessary to type and
underscore between each word of the phrase instead of a
space. The DTC MICRO FILE is an example of the technique.

DTC MICRO FILE DEBUG CONSOLE   Technical
Specification

PURPOSE: The primary use of the Micro File console is to
aid in machine level program development. The
unit will suspend program execution and display
system bus information upon reaching a specific
program location or data transfer condition.
The unit requires no modifications to the Micro
File and simply plugs in for data and power.

INDICATORS: CONTROLS:

    Address Display   Program Restart

    Data Display   R.T.C. Stop

    Memory Write Line   Examine Selector

    Memory Read Line   Program Single Step

    I/O Write Line   Program Continue

    I/O Read Line   Address Trap Set

    Real Time Clock

    Present Interrupt Level

    Wait Condition

    Hold Condition

SIZE: 11" x 9.5" x 6.75" - 27.9cm x 24.1cm x 17.1cm

WEIGHT: 5Lbs - 2.2Kg

Specifications subject to change without notice.

70225

DTC MICRO FILE
PROTOTYPING BOARD

- Fits into mother board of Micro File

- Bus drivers for Micro File bus included

- Over two thousand plated thru holes

- Variety of wire wrap connectors and sockets can be used

- Top connector for ribbon cable output to 'other devices'

- Access to Micro File DC Power

- Price - $250
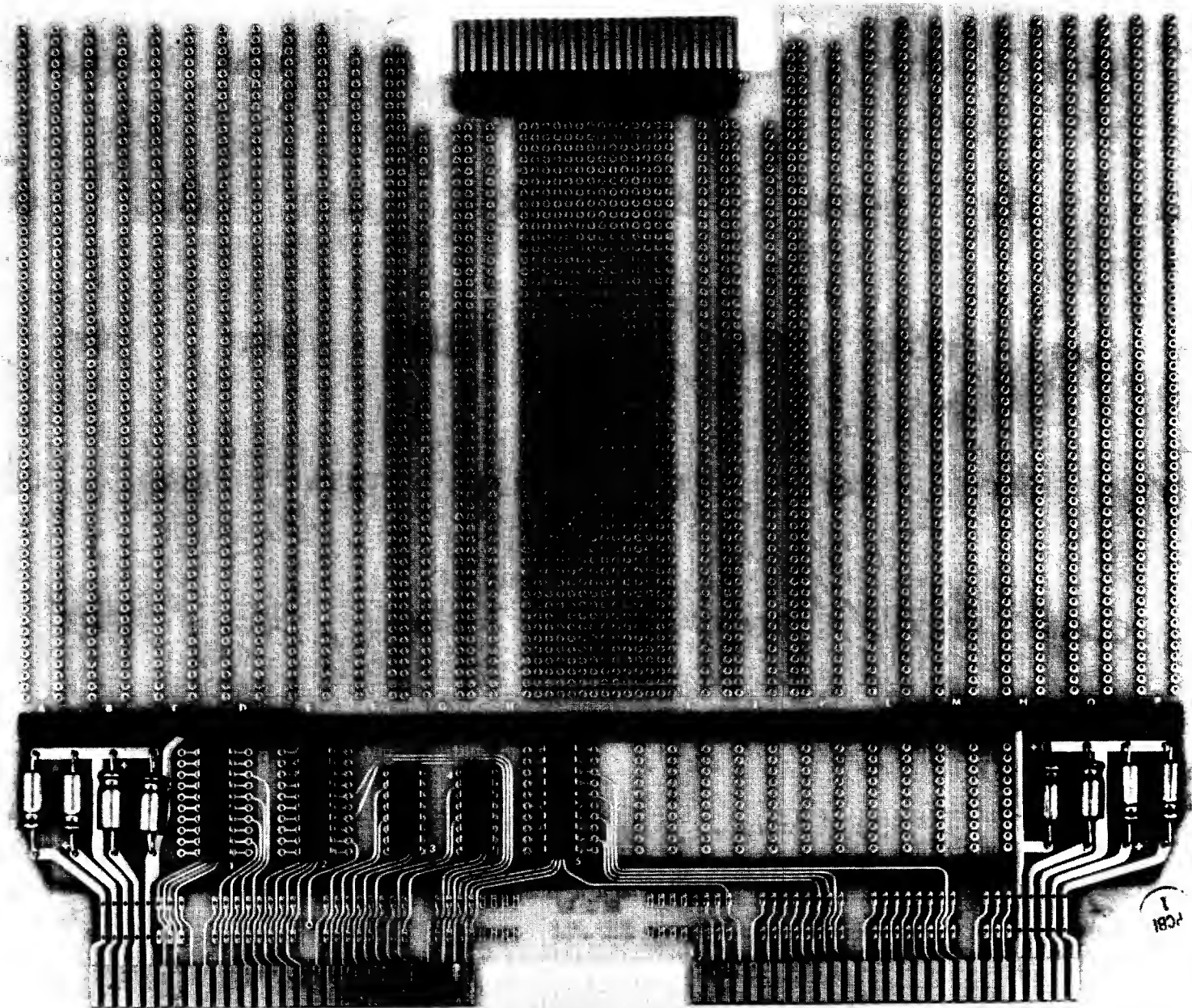
MICRO FILE PARTS LIST

PRINTED CIRCUIT BOARDS

Front Panel - Large	$ 150.00
Front Panel - Small	50.00
Mother Board (4 Slot)	125.00
Mother Board (8 Slot)	175.00
CPU	1,000.00
Disk I/O	500.00
Comm I/O	500.00
Memory 16K (Includes RAM)	1,000.00
Back Panel Assembly	125.00
Disk Drive	1,200.00
Power Supply	425.00
Power Supply - 50 HZ Option	50.00
Front Panel Assembly	50.00
Cables (Each)	25.00
Power Switch Assembly	25.00
Case	125.00

MICRO FILE PRICE LIST

MICRO FILE (Includes 8K RAM, Editor, Letter Writer Program and System Software)	$4,295.00
16K MEMORY BOARD	1,000.00
EXPANDER MOTHER BOARD (Allows total 56K RAM Memory)	175.00
PROGRAMMING AID (Debug Console)	500.00
230V POWER SUPPLY	50.00
CABLES (EIA)	25.00
DISKS (5 Per Pack)	60.00/Pack

SOFTWARE	ONE TIME RENTAL CHARGE
*BASIC	$ 500.00
ASSEMBLER	350.00

*Requires additional 16K on base machine (Total 24K RAM)

I N I T I A L   U S E
# I N S T R U C T I O N S

UNPACKING & SETUP

1 PREPARE A SUITABLE LOCATION FOR YOUR MICRO-FILE AFTER IT IS REMOVED FROM THE SHIPPING CONTAINER   keep in mind the weight and rear panel switches access also try to find an environment that is free of electrical noise created by heavy equipment or motors

2 REMOVE UNIT FROM SHIPPING CONTAINER, VISUALLY INSPECT, AND CHECK THE SERIAL NUMBER AGAINST THAT SHOWN ON THE PACKING LIST OR INVOICE

3 SET THE REAR PANEL SWITCHES FOR DERSIRED DATA TERMINAL EQUIPMENT (DTE) AND DATA COMMUNICATIONS LINE (DCL) OPERATION AND CHARACTERISTICS

4 ALLOW THE UNIT TO REACH ROOM TEMPERATURE (apx. 1 hr)

5 CONNECT DTE AND DCL EQUIPMENT, (DTE should be set to line-interactive mode) CONNECT AC POWER CORD, AND SWITCH ON POWER

6 PUSHING THE MICRO-FILE RESET SWITCH SHOULD PRINT "DTC MICRO FILE" ON THE TERMINAL

7 AT THIS TIME WE STRONGLY SUGGEST YOU MAKE COPIES OF BOTH SYSTEM DISKETTES SHIPPED WITH YOUR MICRO-FILE

OPERATING SUGGESTIONS

1 ALWAYS KEEP BACK-UP DISKETTES OF ALL SYSTEM PROGRAM FILES AND ANY VALUABLE TEXT FILES YOU CREATE

2 USE A GOOD QUALITY DISKETTE TO AVOIDE PREMATURE FAILURE OF THE READ/WRITE HEADS

3 NEVER TRY TO WRITE ON THE "BACK" OF A DISKETTE   THIS MAY DAMAGE THE HEADS

4 YOUR UNIT WAS SHIPPED WITH THE SYSTEM DISKETTE WRITE PROTECTED (open hole) TRY TO KEEP IT AND ANY OTHER VALUABLE DISKETTES PROTECTED AT ALL TIMES

5 JUST AS A MEASURE OF SAFETY WE SUGGEST YOU EJECT THE DISKETTS BEFOR THE SYSTEM POWER IS TURNED OFF

6 GET MAXIMUM PERFORMANCE FROM YOUR NEW MICRO-FILE, READ THE USERS MANUAL AND BECOME FAMILIAR WITH ALL THE COMMANDS AND FEATURES

7 ONCE MORE, KEEP BACK-UP DISKETTES IN A SAFE SECURE PLACE AND ALWAYS USE CARE IN DISKETTE HANDELING

*

HOW TO INSERT MICRO FILE DISKETTE

TOP

Insert
Front Edge
First

O——— Write-Protect*
Hole
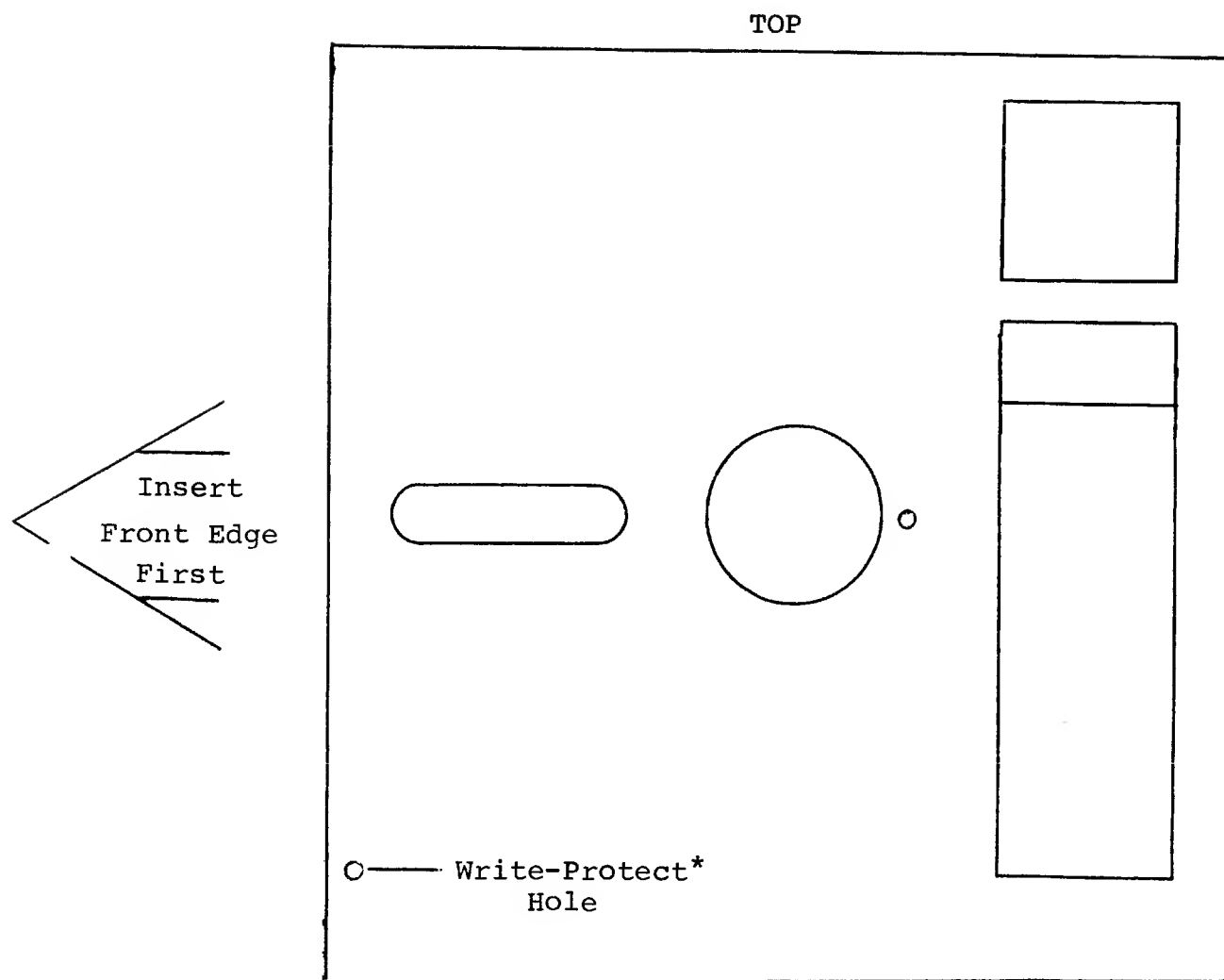
*Cover with silver tape in order to write on diskette.

LIMITED WARRANTY

This Data Terminals and Communications' product is guaranteed to be free of defects in materials and workmanship from date of purchase for a period of 90 days from the date of shipment to the original purchaser. During this period, DTC guarantees repair or replacement (at our option) without charge for parts or labor but excluding transportation charges to and from our repair location designated. In case of repair or replacement, the guarantee period continues from the original date of purchase.

This guarantee is void on products in any way abused, misused, repaired without our authorization, or whose use is contrary to the operating instructions, or on malfunctions due to normal wear and tear. DTC's obligation and liabilities shall be limited to those declared above and in no event shall exceed the original cost of the product thus purchased.

6105

MICRO FILE NEWS

BASIC 2.0

A problem exists in Micro File Basic 2.0 that prevents inputting data from the terminal and then printing it to a sequential file.

EXAMPLE:

```
10 OPEN "O", 1, "FILEA"
20 INPUT A$
30 PRINT #1, A$
40 GOTO 20
```

The above does not work after the first iteration. To correct this problem, first copy your disk with Basic (FOR BACK UP!). Then with Basic on Disk 0 (Write Enabled) type:

```
*LO $BA$
*PA 3D53 C3 23 3E
*PA 6DD5 B1
*ER $BA$ P
*SA $BA$ 2800 6DD9 2800
```

You now have 2.1 Basic.

MICRO FILE NEWS

BASIC 2.2

In order to operate MK IV Micro Files using all four disks from Basic, you need to convert from 2.1 in the following manner:

First copy your disk with Basic (FOR BACK UP), then with Basic on Disk 0 (Write Enabled) type:

```
*LO $BA$
*PA 5A83 04
*PA 6DD5 32
*ER $BA$ P
*SA $BA$ 2800 6DD9 2800
```

You now have 2.2 Basic.

Basic Drops Into Monitor

Should this happen you can get back, in most instances, to Basic without losing your current program by typing:

```
*GO 2800
```

LED Counter Program

Try running the following program and watch the line number
lights on Micro File front panel.

list

```
1ØØ REM Program that will count sequentially
11Ø REM in the front panel display
12Ø DIM I%(4)
13Ø I%(Ø) = I%(Ø)+1
14Ø K% = 8
15Ø FOR J%=Ø TO 3
16Ø IF I%(J%)>9 THEN I%(J%)=Ø : I%(J%+1)=I%(J%+1)+1
17Ø K%=K%+K% : OUT 66,K%+I%(J%)
18Ø NEXT J%
19Ø GOTO 13Ø
```

OK

BASIC 2.3 FEATURES

1. Consistent handling of data INPUT from terminal or files. (See write up).

2. CR alone on INPUT gives zero or null string, but does not stop the program.

3. BREAK (ATTN) is the only key that will stop the program. C^C and O^C no longer work. Type-ahead now works on input. Stopping the program this way stops typeout (to the terminal and line) and throws out any input to the terminal.

4. Support of 4 disks.

5. Correction of a bug in PRINT #f

6. Correction of a bug in LCHR$

7. Null statements are allowed.

   (ex: IF X<5 THEN ˄ ELSE 1250

                       null statement

   or   X=5: ˄ : Y=4
           null statement

   or   'comment as the only text on a line).

8. A REM line can be written with a single quote instead.

   ex: 940 'COMMENT

9. Long programs will be faster because of improved processing in GOTOs and GOSUBs, and faster number conversion for floating point.

10. You can now load BASIC and start a file with a single command:

BA fn [Dn]   CR

This is equivalent to:

BA   CR

LOAD "fn",dn,R   CR

The normal BASIC header lines are not printed if a file name is given.

11. DEL #f on sequential files now releases empty sectors.

Conventions for READ and INPUT data

1. Numeric Input:

- A numeric item may have a leading + or - sign (unless it is an octal value).

- Numeric items are terminated by blank, comma, or end-of-line. This is a change from the previous version.

  Example:

     1 2 3 is the same as 1,2,3 now, but it was the same as 123 in earlier versions of DTC BASIC.

- Either a comma or spaces (or both) may be used to separate items. A null item (e.g: '''',,'''') will be input as zero.

2. String Input

- Unquoted strings begin after leading blanks and continue until a comma or end-of-line. Trailing blanks are preserved. Leading control characters (such as BEL) are now preserved as part of the string.

- Quoted strings terminate at the matching quote. For non-disk INPUT or READ, the end-of-statement will also terminate a quoted string.

- String items may be separated by a comma or spaces. A null item will be input as the null string.

MICRO FILE NEWS

BASIC 2.4
New Text Editor
LETP

BASIC 2.4 is now released. It supersedes 2.3 (few copies of which
were distributed). Attached is a sheet describing the features.

A new version of the Text Editor is released that allows:

1. New Commands: L, LN, LX
    These are like P, PN, PX except that output is to the line
    EIA port (DCL).

2. During MODIFY a backspace typed in while at the leftmost
    line position will move the carriage one place left (but
    otherwise has no effect).

3. String replacement with a count now continues from the point
    of last change, rather than from the beginning of the current
    line.

    This means that (e.g.) 'X'='XX'25 now works the way you
    want it to.

    Note that this only happens with:

        'string-1'='string-2' n

    and not with

        'string-1'='string-2'   CR

        An   CR

    Since the interior line positioning is lost once the command
    is first exited.

    (example: If a line is: ABCDABCDA then 'A'='AX'3 CR
               will give: AXBCDAXBCDAX and not: AXXXBCDABCDA as
               before).

You can find out the version of the Editor you are using by typing
a ? at the - prompt level.

LETP is a version of the Letter Writer that provides 'paging with
page numbers'. Set to 54 lines per page, LETP will automatically
form text into numbered pages. It is run in the same manner as LET.

Registered Software Update Service users will be receiving these
packages shortly.

DTC BASIC reference cards are being printed, and a complimentary
copy will be mailed soon. Additional copies will be 50¢. We are
still editing the new version of the DTC BASIC manual, but are
pushing to get it out as fast as possible.

61215

MICRO FILE

DTC BASIC 2.4 FEATURES

1. Consistent handling of data INPUT from terminal or files. (See write up).

2. CR alone on INPUT gives zero or null string, but does not stop the program.

3. BREAK (ATTN) is the only key that will stop the program. C^C and O^C no longer work. Type-ahead now works on input. Stopping the program this way stops typeout (to the terminal and line) and throws out any input to the terminal.

4. Support of 4 disks.

5. Correction of a bug in PRINT #f

6. Correction of a bug in LCHR$

7. Null statements are allowed.

   (ex: IF X<5 THEN ↑ ELSE 1250
                    null statement

   or   X=5:↑: Y=4
         null statement

   or   'comment as the only text on a line).

8. A REM line can be written with a single quote instead.

   ex: 940 'COMMENT

9. Long programs will be faster because of improved processing in GOTOs and GOSUBs, and faster number conversion for floating point.

10. You can now load BASIC and start a file with a single command:

        BA fn [Dn]   CR

    This is equivalent to:

        BA   CR

        LOAD "fn",dn,R   CR

    The normal BASIC header lines are not printed if a file name is given.

11. DEL #f on sequential files now releases empty sectors.

Conventions for READ and INPUT data

1. Numeric Input:

    • A numeric item may have a leading + or - sign
      (unless it is an octal value).

    • Numeric items are terminated by blank, comma,
      or end-of-line. This is a change from the
      previous version.

      Example:

        1 2 3 is the same as 1,2,3 now, but it was
        the same as 123 in earlier versions of DTC
        BASIC.

    • Either a comma or spaces (or both) may be used
      to separate items. A null item (e.g: '''',,'''')
      will be input as zero.

2. String Input

    • Unquoted strings begin after leading blanks and
      continue until a comma or end-of-line. Trailing
      blanks are preserved. Leading control characters
      (such as BEL) are now preserved as part of the
      string.

    • Quoted strings terminate at the matching quote.
      For non-disk INPUT or READ, the end-of-statement
      will also terminate a quoted string.

    • String items may be separated by a comma or spaces.
      A null item will be input as the null string.

61215

System Interface Section

The Terminal Input buffer is located at 8223 (201F-hex). The Line Input buffer is located at 8485 (2125-hex). See the Programmer's Reference Manual for a description of how these buffers are used.

Useful Functions:

1) Terminal Input buffer has data:

   PEEK(8224)<>PEEK(8225)

2) Line Input buffer has data:

   PEEK(8486)<>PEEK(8487)

3) Last character of Terminal Input is CH (set Y ≠ 0)

   ```
   1000 Y = PEEK(8224)<>PEEK(8225)
   1010 IF Y THEN
            IF PEEK(8225) THEN
               Y = PEEK(8225+PEEK(8225))=CH ELSE
               Y = PEEK(8225+PEEK(8223))=CH
   1020 RETURN
   ```

4) Last character of Line Input is CH (set Y ≠ 0)

   ```
   2000 Y = PEEK(8486)<>PEEK(8487)
   2010 IF Y THEN
            IF PEEK(8487) THEN
               Y = PEEK(8487+PEEK(8487))=CH ELSE
               Y = PEEK(8487+PEEK(8485))=CH
   2020 RETURN
   ```

   NOTE: Multi-line indented statements can be created by typing: (e.g.)

   IF Y THEN (LF)(CR)(SP)(SP)(SP)(SP) IF PEEK...
                            ↑
                      TRANSPARENT switch "ON" when typing this.

5) Minimum of two numbers (A,B)

   -((A<=B)*A+(B<A)*B)

6) Maximum of two numbers (A,B)

   -((A>=B)*A+(B>A)*B)

MICRO FILE NEWS - 70608

FORTRAN

Two companies have approached DTC with Fortran Compilers
that they are willing to convert to Micro File. Both
claim speeds of execution up to 30 times that of Basic!

We would like to hear from users as to whether they are
interested in such a package, and whether you would
consider purchasing it for $500 a copy.

APL and COBOL

One of the two companies mentioned above has APL 70%
complete and has started on COBOL. If you are interested
in such packages, we would appreciate hearing from you.

Your response, to I. Sandoval, Marketing Coordinator, in
the next 10 days would really be appreciated

MICRO FILE BASIC

A string is either referred to in the form A$ (the $ sign
designates it as a string) or as a literal string between
double quote marks (").

This applies to Basic programs and strings used as part of
a MON command.

    e.g.   MON "FL D1"
             and
             A$ = "FL D1"
             MON A$

             Will both result in a file listing of disk 1

MICRO FILE NEWS LETTER - 70830

EXTENDED BASIC 3.2

EXTENDED BASIC 2.4 is now being replaced by a new version; EXTENDED BASIC 3.2. There are corrections for operational problems with some variations of instructions in EXTENDED BASIC 2.4. The impressive news with EXTENDED BASIC 3.2 is incorporation of new instructions, Expanded functions on existing instructions and extensive program error reporting capability.

I.   Basic 3.2 incorporates fixes for "bugs" in Basic 2.4 as follows:

1.   The LOF (file number) input output function for returning the highest record number written to a random file would give inconsistent results if the record number was a multiple of 64.

2.   Number syntax has been tightened. Previously, an expression like "1D +2" would yield a result of 100, (1D+2) rather than the correct result of 3,(1D)+(2).

3.   A similar syntax error in the FIELD statement caused "FIELD #1,N as A$" to give a syntax error. The space after N was not being recognized.

4.   An "Out of Space" condition is now detected, if the disk is full, when writing to disk using data I/O within BASIC.

5.   The "bug" in the decoding of expressions in the form "AB"="A"+"B", is now fixed.

II. Improvements and New Features in Basic 3.2

It is important to note that Basic programs created in Basic 2.4 must be SAVEd as a text file then TLOADed in Basic 3.2. The program can now be SAVEd as a binary file.

1. Chaining programs using "LOAD...,R" will preserve ALL variables, strings, arrays, etc. unless there is not enough space for the load.

2. The characters [ ] will be allowed as subscript delimiters as well as ( ).

3. DEFINT,-STR,-SNG,-DBL now all allow multiple ranges: e.g. DEFINT A,E,I,-O,U.

4. READ or INPUT into a double precision variable forces double precision conversion.

5. In addition to &nnn for octal constants, the following new forms are allowed: &Onnn (octal), &Hnnn (hexadecimal), &Bnnn (binary).

6. The following new conversions from integer to string (like STR$) are available: OCT$(val), HEX$(val), BIN$(val).

7. The functions PEEK,POKE,HEX$,OCT$, and BIN$ can have arguments in the range -32768, = val, =+65535.

8. The commands DELETE x-y and DELETE x- now works even when line number "x" does not exist. The next higher numbered line is taken.

9. The REN number statement now has default values of 10,10 instead of 100,10.

10. A new command, STEP, is like TRON except that execution pauses after typing the statement number and before the statement is executed. Typing any non-break character allows the statement to execute. To exit the STEP function, depress break then enter TROFF carriage return.

11. The MODulus function is redefined to give correct mathematical results for negative arguments, thus: 7 MOD4=3,-7 MOD -4=-3, -7 MOD 4=1, 7 MOD -4=-1.

12. New arithmetic operators on integers: XOR (exclusive OR), EQV (equivalence), and IMP (implication); have precedence just below "OR". Definitions are:

    A XOR B=(A and Not B) or (B and Not A)
    A EQV B=(A and B) or (Not A and Not B)=Not (A XOR B)
    A IMP B=(Not A) or B

13. New operators MIN and MAX work on all numeric data types: 3 MIN 7=3

14. New operators to perform shifts on 16 bit integers. SHL, SHR, ROL, ROR:

    A SHL B shifts A left B bits, zeros shifted
            into right end
    A SHR B shifts A right B bits, the sign bit
            is extended to the right
    A ROL B rotates A left B bits, bits shifted
            off the left end cycle into the right end
    A ROR B rotates A right B bits, bits shifted
            off the right end cycle into the left end

15. A revised complete list of numeric operator precedences is:

    ¢   (exponentiation)
    * and /

    MOD
    + and -
    MIN and MAX
    SHL, SHR, ROL, ROR
    =, ,., ,=, .= (relationals)
    NOT(this and following operators work on
        integers only)
    AND
    OR
    XOR
    EQV
    IMP

16. Typing BREAK during string compaction will stop compaction with the message:

    BREAK IN STRING COMPACTION

    Break can be used to stop a TLOAD operation.

17. A user may have up to 16 files simultaneously active (numbers 1..16). This is a new implementation with the following features:

> File buffers are allocated when needed, OUT OF STRING SPACE, so string space declarations (CLEAR nn) should allow for 128 bytes per file.

> A file may be open for INPUT simultaneously under more than one file number. OUTPUT on a file that is open as another number is not protected, and can cause severe problems. Users will have to watch for this on their own.

18. The statement RANDOMIZE has been added. It is functionally equivalent to ... = RND(-PEEK(8197)), etc.

19. AUTO nn,ii statement begins auto input with line "nn", and increment "ii". Defaults are 10,10.

20. For AUTO, DELETE, LIST and EDIT, the symbol "." stands for the current line. (So you can say "DELETE.").

21. The function calls LEFT$(A$,0) and RIGHT$(A$,0) are now legal and will return the null string.

22. New functions SPACE$(size) and STRING$(chr$, size) return strings of length (size) and all spaces or "chr".

23. New functions TRIML$(string), TRIMR$(string), and TRIM$(string), that remove leading spaces (L), trailing spaces (R) or both leading and trailing spaces.

24. New function LABEL$(disk number) returns the disk label as a string.

25. New function VARPTR(variable or file number((n)) will return the address of the variable, or the address of the string descriptor, or the address of the file control block for the specified file. Useful for communicating with assembly language subroutines.

4

26. New function DELAY(seconds (file)), will delay up to 10^{38} seconds, or until there is input (not necessarily a full line) for the specified file. The file number must be 98(DTE) or 99 (DCL). If no file number is specified, DTE is assumed. If the seconds timer runs out, DELAY returns -1 (TURE), otherwise it returns 0 if data exists. Because of variations in individual Micro Files, the timer is only accurate to within 5-10 percent.

27. New command SLEEP will shut off all disk motors until disk activity or a MON request or BYE restarts them. A null MON request will restart all disks, e.g. MON''''

28. New command UNLOAD dn (dn)...will eject the specified disks if the Micro File is equiped for this option. UNLOAD with no parameters ejects all disks.

29. MID$(...) is allowed on the left side of an assignment statement, and will cause replacement of characters interior to a string without requiring copying the string. This should be a good speedup feature. e.g. MID$(A$,4,1)="X" instead of A$=LEFT$(A$,3) + "X" + MID$(A$,5)

30. The LINE INPUT command allows a prompt, similar to INPUT, e.g. LINE INPUT "Type input line"; A$

31. File #98 defaults to DTE, #99 defaults to DCL, to allow LCHR$ from DTE, and uniform PRINT, INPUT statements with variable file numbers for debug to terminal.

ERROR AND BREAK DETECTION AND PROCESSING IN BASIC 3.2

Errors and breaks typed from the terminal may now be trapped within BASIC. The user may now detect errors and provide recovery action before returning to the program. The statements that make this possible are ON ERROR GOTO, RESUME, and ERROR, along with the ERR and ERL variables. In the following description BREAK may replace ERROR.

1. Enabling Error Trapping. The ON ERROR GOTO statement specifies the line of the DTC BASIC program on which the error handling subroutine starts. The format is as follows:

        ON ERROR GOTO line number

The ON ERROR GOTO statement should be executed before the user expects any errors to occur. Once an ON ERROR GOTO statement has been executed, all errors detected will cause BASIC to start execution of the specified error handling routine.

2. Disabling error trapping. ON ERROR GOTO 0 disables trapping of errors so any subsequent error will cause BASIC to print an error message and stop program execution. If an ON ERROR GOTO 0 statement appears in an error trapping subroutine, it will cause BASIC to stop and print the error message that caused the trap. It is recommended that all error trapping subroutines execute an ON ERROR GOTO 0 if an error is encountered for which they have no recovery action.

NOTE: If an error occurs during the execution of an error trap routine, the system error message will be printed and execution will be terminated. Error trapping does not trap errors within the error trap routine. Breaks in a break routine cause a conventional message.

3. The ERR and ERL variables. When the error handling subroutine is entered, the variable ERR contains the error code for the error. (NOTE: You can produce a list of error codes for publication by repeated execution of the ERROR statement, see below).

The ERL variable contains the line number of the line where the error or break occurred. If the statement that caused the error was a direct (command) mode statement, ERL will be equal to 65535. To test if some particular line was involved in an error, use the statement form:

        IF ERL = line number THEN...

This form, with ERL on the left of an "=" allows the statement number to be renumbered with the REN command properly.

4. The RESUME statement. The RESUME statement is used to continue execution of the BASIC program after the error recovery procedure has been performed. The user has three options. The program may be RESUMEd at the statement that caused the error (re-executing the statement), the program may be RESUMEd at the statement immediately following the error, or the program may be RESUMEd at a particular line. To RESUME at the statement that caused the error use:

        RESUME
or
        RESUME 0

To RESUME at the statement following the error, use:

        RESUME NEXT

To RESUME at an arbitrary line, use:

        RESUME line number

The following example shows how a simple error trapping subroutine operates:

```
100 ON ERROR GOTO 500
200 INPUT "What are the numbers to divide? "; X,Y
210 Z=X/Y
220 PRINT "Quotient is"; Z
230 GOTO 200
500 IF ERR=12 AND ERL=210 THEN 520
510 ON ERROR GOTO 0
520 PRINT "You can't have a divisor of zero!"
530 RESUME 200
```


5. The ERROR statement. In order to force branching to an error trapping routine, an ERROR statement has been provided. The primary use of the ERROR statement is to allow the user to define his own error codes which can then conveniently be handled by a centralized error trap routines as described above. The format of the ERROR statement is:

        ERROR integer expression

When defining error codes, values should be picked which are greater than the ones used by BASIC. The maximum error number

7

is 255. Of course, the error statement may also be used to force any standard error. Use of an ERROR statement to force printout of an error message for which no error text is defined will cause an UNPRINTABLE ERROR error message to be printed.

8

DA20-21 = USART #1

DA20 = DATA I/O
DA21 = STATUS I/O

For more detail on the USART assignments, refer to Intel specifications (8251A)

	I/O READ			I/O WRITE	
DA22	DB$\emptyset$ = $\overline{R\emptyset}$			DB$\emptyset$ = A	
	DB1 = $\overline{R1}$	BAUD RATE		DB1 = B	BAUD RATE
	DB2 = $\overline{R2}$	(USART#1)		DB2 = C	USART#1
	DB3 = UNASSIGNED			DB3 = D	
	DB4 = $\overline{P\emptyset}$			DB4 = DO NOT USE	
	DB5 = $\overline{P1}$	PARITY		DB5 = DO NOT USE	
	DB6 = $\overline{P2}$	(USART#1)		DB6 = DO NOT USE	
	DB7 = UNASSIGNED			DB7 = DO NOT USE	
DA23	DB$\emptyset$ = $\overline{HALF}$			DB$\emptyset$ = TRANS. INT. CLEAR (USART#1)	
	DB1 = $\overline{ECHO}$	$\overline{DUPLEX}$		DB1 = TRANS. INT. CLEAR (USART#2)	
	DB2 = $\overline{FULL}$	(USART#1)		DB2 = TRANS. INT. CLEAR (USART#3)	
	DB3 = $\overline{\emptyset}$			DB3 = TRANS. INT. CLEAR (MPCC)	
	DB4 = $\overline{5}$	DELAY		DB4 = UNASSIGNED	
	DB5 = $\overline{10}$	(USART#1)		DB5 = UNASSIGNED	
	DB6 = $\overline{20}$			DB6 = UNASSIGNED	
	DB7 = UNASSIGNED			DB7 = UNASSIGNED	

DA24-25 = USART #2

DA24 = DATA I/O
DA25 = STATUS I/O

For more detail on the USART
assignements, refer to Intel
specifications (8251A).

I/O READ   I/O WRITE

DA26   DB$\emptyset$ = $\overline{R\emptyset}$ DB$\emptyset$ = A

        DB1 = $\overline{R1}$ ⎫ BAUD RATE   DB1 = B   ⎫ BAUD RATE
                                                                        ⎬   USART #2
        DB2 = $\overline{R2}$ ⎭ (USART#2)   DB2 = C   ⎭

        DB3 = UNASSIGNED   DB3 = D

        DB4 = $\overline{P\emptyset}$ DB4 = DO NOT USE

        DB5 = $\overline{P1}$ ⎫ PARITY   DB5 = DO NOT USE

        DB6 = $\overline{P2}$ ⎭ (USART#2)   DB6 = DO NOT USE

        DB7 = UNASSIGNED   DB7 = DO NOT USE

DA27   DB$\emptyset$ = $\overline{HALF}$ DB$\emptyset$ = UNASSIGNED

        DB1 = $\overline{ECHO}$ ⎫ DUPLEX   DB1 = UNASSIGNED

        DB2 = $\overline{FULL}$ ⎭ (USART#2)   DB2 = UNASSIGNED

        DB3 = $\overline{\emptyset}$ DB3 = UNASSIGNED

        DB4 = $\overline{5}$ ⎫ DELAY   DB4 = UNASSIGNED

        DB5 = $\overline{1\emptyset}$ ⎭ (USART#2)   DB5 = UNASSIGNED

        DB6 = $\overline{2\emptyset}$ DB6 = UNASSIGNED

        DB7 = UNASSIGNED   DB7 = UNASSIGNED

```
                          S1000
              ASYNCHRONOUS I/O CONTROL
                          PORTS


DA28-29 = USART #3

DA28 = DATA I/O                        For more detail on the USART
DA29 = STATUS I/O                      assignments, refer to Intel
                                       specifications (8251A).


            I/O READ                   I/O WRITE


DA2A     DBØ = $\overline{R\emptyset}$         ⎫         DBØ = A         ⎫
                                       ⎬ BAUD RATE                      ⎬ BAUD RATE
         DB1 - $\overline{R1}$         ⎬                 DB1 = B         ⎬
                                       ⎭ (USART #3)                      ⎬  USART #3
         DB2 = $\overline{R2}$                 (USART #3)       DB2 = C         ⎭
                                                         DB3 = D
         DB3 = UNASSIGNED

         DB4 = $\overline{P\emptyset}$         ⎫         DB4 = DO NOT USE
                                       ⎬ PARITY
         DB5 = $\overline{P1}$         ⎬                 DB5 = DO NOT USE
                                       ⎭ (USART #3)
         DB6 = $\overline{P2}$                           DB6 = DO NOT USE

         DB7 = SYSTEM BIT                        DB7 = DO NOT USE


DA2B     DBØ = $\overline{HALF}$        ⎫        DBØ = UNASSIGNED
                                       ⎬ DUPLEX
         DB1 = $\overline{ECHO}$        ⎬                DB1 = UNASSIGNED
                                       ⎭ (USART #3)
         DB2 = $\overline{FULL}$                DB2 = UNASSIGNED

         DB3 = $\overline{\emptyset}$          ⎫        DB3 = UNASSIGNED

         DB4 = $\overline{5}$           ⎬ DELAY         DB4 = UNASSIGNED

         DB5 = $\overline{1\emptyset}$          ⎬ (USART #3)     DB5 = UNASSIGNED

         DB6 = $\overline{2\emptyset}$          ⎭        DB6 = UNASSIGNED

         DB7 = UNASSIGNED                       DB7 = UNASSIGNED
```

S1000
ASYNCHRONOUS COMMUNICATIONS
THIRD I/O PORT SWITCH SETTINGS

DA2A	BAUD	1	2	3		PARITY	4	3	2
LOC. A13	110	OFF	OFF	OFF	LOC. A14	EVEN	OFF	OFF	OFF
	300	ON	OFF	OFF		ODD	ON	OFF	OFF
	1200	OFF	ON	OFF		1	OFF	ON	OFF
	2400	ON	ON	OFF		0	ON	ON	OFF
	4800	OFF	OFF	ON		NONE	OFF	OFF	ON
	9600	ON	OFF	ON					

OPEN = OFF

DA2B	DUPLEX	1	2		DELAY	3	4
LOC. 27	ECHO	OFF	OFF	LOC. 27	5/10	OFF	OFF
	FULL	ON	OFF		0	ON	OFF
	HALF	OFF	ON		20	OFF	ON

DA2A   SYSTEM BIT = LOC. A14, SWITCH 1

LOGIN FILE
SWITCH

S1000
USART BAUD RATE CONVERSION
ROM CODE

ADDRESS	CODE	COMMENTS
00	FF	CONVERT FROM EXT. TO 19,200
01	00	CONVERT TO 50
02	11	CONVERT TO 75
03	22	CONVERT TO 110
04	33	CONVERT TO 134.5
05	44	CONVERT TO 150
06	55	CONVERT TO 300
07	66	CONVERT TO 600
08	99	CONVERT FROM 900 TO 2000
09	77	CONVERT TO 1200
0A	88	CONVERT TO 1800
0B	AA	CONVERT TO 2400
0C	BB	CONVERT TO 3600
0D	CC	CONVERT TO 4800
0E	DD	CONVERT TO 7200
0F	EE	CONVERT TO 9600
10	00	PASS 50
11	11	PASS 75
12	22	PASS 110
13	33	PASS 134.5
14	44	PASS 150
15	55	PASS 300
16	66	PASS 600
17	77	PASS 1200
18	88	PASS 1800
19	99	PASS 2000
1A	AA	PASS 2400
1B	BB	PASS 3600
1C	CC	PASS 4800
1D	DD	PASS 7200
1E	EE	PASS 9600
1F	FF	PASS 19,200

S1000
COM 5016
BAUD RATE CODE TABLE

BAUD RATE	D	C	B	A
50	0	0	0	0
75	0	0	0	1
110	0	0	1	0
134.5	0	0	1	1
150	0	1	0	0
300	0	1	0	1
600	0	1	1	0
1200	0	1	1	1
1800	1	0	0	0
2000	1	0	0	1
2400	1	0	1	0
3600	1	0	1	1
4800	1	1	0	0
7200	1	1	0	1
9600	1	1	1	0
19,200	1	1	1	1

SYNC OPTION I/O BOARD
SWITCH SETTINGS

PORT 2

DA 2A
SW LOC 13

Baud Rate	1	2	3
110	off	off	off
300	on	off	off
1200	off	on	off
2400	on	on	off
4800	off	off	on
9600	on	off	on

DA 2A
SW LOC 14

Parity	4	3	2
Even	off	off	off
Odd	on	off	off
1	off	on	off
0	on	on	off
Hex None	off	off	on

DA 2A
SW LOC 23

Duplex	3	4
Echo	off	off
Half	on	off
Full	off	on

DA 2B
SW LOC 23

Delay	1	2
5/10*	off	off
0	on	off
20	off	on

* Jumper X to E for 5

" X to F for 10

SYSTEMS BIT = DA 2A Switch Loc 14 SW 1. When in the OFF
position will EXECUTE a T file named LOGIN.

SYSTEMS DISK SELECT = DA 2A Switch Loc 13, SW 4   . When
in the ON position selects Floppy D0,
in the OFF position selects Hard D4 as
the PWR- on systems disk.   7

PORT 3
SYNC PORT

DA 2F
SW LOC 9

	1	2	3	4
	*	*	*	UNUSED

* Protocall definiton
switches for sync.
intovface

TASKMASTER

ASYNCHRONOUS OPTION BOARD
SWITCH SETTINGS

PORT 2

DA 2A
SW LOC 12

Baud Rate	1	2	3
110	off	off	off
300	on	off	off
1200	off	on	off
2400	on	on	off
4800	off	off	on
9600	on	off	on

DA 2A
SW LOC 13

Parity	1	2	3
Even	off	off	off
Odd	on	off	off
1	off	on	off
0	on	on	off
Hex None	off	off	on

DA 2A
SW LOC 15

Deplex	1	2
Echo	off	off
Half	on	off
Full	off	on

DA 2A
SW LOC 15

Delay	3	4
5/10*	off	off
0	on	off
20	off	on

SYSTEM BIT = DA 2A Switch Loc 13, sw 4. When in the (off) *CAP*
            position will EXECUTE a T file named LOGIN.

SYSTEM DISK SELECT = DA 2A Switch Loc 12, sw 4. When in *CAI* the (on) position selects Floppy D0, in the (off) position selects Hard D# as the PWR -/ on system disk.   **?**
            CAP

* Jumper   X to B   for 5 on Port 2
    "   X to A   for 10 on Port 2
    "   X to D   for 5 on Port 3
    "   X to C   for 10 on Port 3

TASKMASTER

ASYNCHRONOUS OPTION BOARD
SWITCH SETTINGS

PORT 3

DA 2E
SW LOC 10

Baud Rate	1	2	3
110	off	off	off
300	on	off	off
1200	off	on	off
2400	on	on	off
4800	off	off	on
9600	on	off	on

DA 2E
SW LOC 11

Parity	1	2	3
Even	off	off	off
Odd	on	off	off
1	off	on	off
0	on	on	off
Hex None	off	off	on

DA 2F
SW LOC 14

Duplex	1	2
Echo	off	off
Half	on	off
Full	off	on

DA 2F
SW LOC 14

Delay	3	4
5/10*	off	off
0	on	off
20	off	on

SYSTEM BIT = DA 2A Switch Loc 13, SW 4. When in the OFF
     position will EXECUTE a T file, named LOGIN.

SYSTEM DISK SELECT = DA 2A Switch Loc 12, SW 4. When
                     in the ON position selects Floppy
                     D0, in the OFF position selects
                     Hard D as the PWR- on systems disk.

                     7

    * Jumper   X to B for 5 on Port 2
         "   X to A for 10 on Port 2
         "   X to D for 5 on Port 3
         "   X to C for 10 on Port 3

	PIN#	FUNCTION			PIN#	FUNCTION	
	1	EARTH GROUND			1	EARTH GROUND	
	2	DATA TO MKX	(T)		2	DATA FROM MKX	(F)
	3	DATA FROM MKX	(F)		3	DATA TO MKX	(T)
	4	*REQUEST TO SEND	(T)		4	REQUEST TO SEND	(F)
PORT 0	5	CLEAR TO SEND	(F)	PORT 0	5	*CLEAR TO SEND	(T)
TO	6	DATA SET READY	(F)	TO	6	DATA SET READY	(T)
TERM	7	SIGNAL GROUND		MODEM	7	SIGNAL GROUND	
	9	+12 V. FOR TESTING			9	+12 V. FOR TESTING	
	20	DATA TERMINAL READY	(T)		20	DATA TERMINAL READY	(F)
	1	EARTH GROUND			1	EARTH GROUND	
	2	DATA TO MKX	(T)		2	DATA FROM MKX	(F)
	3	DATA FROM MKX	(F)		3	DATA TO MKX	(T)
	4	*REQUEST TO SEND	(T)		4	REQUEST TO SEND	(F)
PORT 1	5	CLEAR TO SEND	(F)	PORT 1	5	*CLEAR TO SEND	(T)
TO	6	DATA SET READY	(F)	TO	6	DATA SET READY	(T)
TERM	7	SIGNAL GROUND		MODEM	7	SIGNAL GROUND	
	9	+12 V. FOR TESTING			9	+12 V. FOR TESTING	
	20	DATA TERMINAL READY	(T)		20	DATA TERMINAL READY	(F)
	1	EARTH GROUND			1	EARTH GROUND	
	2	DATA TO MKX	(T)		2	DATA FROM MKX	(F)
	3	DATA FROM MKX	(F)		3	DATA TO MKX	(T)
	4	*REQUEST TO SEND	(T)		4	REQUEST TO SEND	(F)
PORT 2	5	CLEAR TO SEND	(F)	PORT 2	5	*CLEAR TO SEND	(T)
TO	6	DATA SET READY	(F)	TO	6	DATA SET READY	(T)
TERM	7	SIGNAL GROUND		MODEM	7	SIGNAL GROUND	
	9	+12 V. FOR TESTING			9	+12 V. FOR TESTING	
	20	DATA TERMINAL READY	(T)		20	DATA TERMINAL READY	(F)

NOTES:

* IF THIS PIN IS USED THE SIGNAL MUST BE TRUE (+) BEFORE THE MKX WILL TRANSMIT DATA. (THIS LINE MAY BE USED TO INTERRUPT DATA TRANSMISSION)

1. ONLY PINS 2,3,&7 NEED BE USED FOR OPERATION MKX

2. (T)=TO MKX   (F)=FROM

3. THE ABOVE I/O PORTS HAVE A CONFIGURATION SWITCH TO CORRECT PIN CONNECTIONS FOR TERMINAL AND MODEM INTERRACES.

PIN#	FUNCTION			PIN#	FUCTION	
	1 EARTH GROUND				1 EARTH GROUND	
	2 DATA FROM MKX	(F)			2 DATA FROM MKX	(F
	3 DATA TO MKX	(T)			3 DATA TO MKX	(T
PORT 3	4 REQUEST TO SEND	(F)		PORT 3	4 REQUEST TO SEND	(F
ASYNC OPTION)	5 *CLEAR TO SEND	(T)		(SYNC OPTION)	5 CLEAR TO SEND	(T
	6 DATA SET READY	(T)			6 DATA SET READY	(T
	7 SIGNAL GROUND				7 SIGNAL GROUND	
	9 +12 V. FOR TESTING				8 CARRIER DETECT	(T
	20 DATA TERMINAL READY	(F)			11 TX REVERSE CHANEL	(F
					12 SPEED MODE	(F
					15 TRAMSMITT CLOCK	(T
					17 RECIEVE CLOCK	(T
					20 DATA TERMINAL READY	(F
					22 RING INDICATOR	(T

PORT 3
(Mailmaster
Option)

1 EARTH GROUND
2 (TAASEL) TWX ACESS SELECT
4 (+V) +V REFERENCE
5 (OH) OFF HOOK
6 (CCT) COUPLER CUT THROUGH
7 SIGNAL GROUNG
8 (DA) DATA TRANSMISSION
11 (RIO) RING INDICATOR 0
12 (RI1) RING INDICATOR 1
17 (DT) DATA TIP
20 (-V) -v REFERENCE
22 (DR) DATA RING

NOTE: * IF THIS PIN IS USED THE SIGNAL
MUST BE TRUE (+) BEFORE THE MKX
WILL TRANSMIT DATA. (THIS LINE
MAY BE USED TO INTERRUPT DATA
TRANSMISSION)

1. ONLY PINS 2,3,&7 NEED BE USED
FOR OPERATION

2. (T)=TO MKX   (F)=FROM MKX

EIA RS232B/C PIN ASSIGNMENTS

1 AA PROTECTIVE GROUND

2 BA TRANSMIT DATA

3 BB RECEIVE DATA

4 CA REQUEST TO SEND

5 CB CLEAR TO SEND

6 CC DATA SET READY

7 AB SIGNAL GROUND

8 CF CARRIER DETECT

9 +P +V. FOR TESTING

10 -P -V. FOR TESTING

11 CY ORIGINATE MODE

12 CX LOCAL MODE

13-19 UNASSIGNED

20 CD DATA TERMINAL READY

21 UNASSIGNED

22 CE RING INDICATOR

23-24 UNASSIGNED

25 CN TERMINAL BUSY

MICRO FILE INTERFACE PIN ASSIGNMENTS

	PIN#	FUNCTION	
	1	EARTH GROUND	
	2	DATA TO MICRO FILE	(T)
	3	DATA FROM MICRO FILE	(F)
	4	*REQUEST TO SEND	(T)
D.T.E.	5	CLEAR TO SEND	(F)
	6	DATA SET READY	(F)
	7	SIGNAL GROUND	
	9	+12 V. FOR TESTING	
	20	DATA TERMINAL READY	(T)

	PIN#	FUNCTION	
	1	EARTH GROUND	
	2	DATA FROM MICRO FILE	(F)
	3	DATA TO MICRO FILE	(T)
	4	REQUEST TO SEND	(F)
D.C.L.	5	*CLEAR TO SEND	(T)
	6	DATA SET READY	(T)
	7	SIGNAL GROUND	
	9	+12 V. FOR TESTING	
	20	DATA TERMINAL READY	(F)

NOTES:

* IF THIS PIN IS USED THE SIGNAL MUST BE TRUE (+) BEFORE THE MICRO FILE WILL TRANSMIT DATA. (THIS LINE MAY BE USED TO INTERRUPT DATA TRANSMISSION)

1 ONLY PINS 2,3,&7 NEED BE USED FOR OPERATION

2 (T)=TO MICRO FILE   (F)=FROM MICRO FILE

3 IF THE D.C.L. PORT IS CONFIGURED AS A D.T.E. PORT BY INTERNAL JUMPER, USE D.T.E. ASSIGNMENTS TABLE

EIA RS232B/C PIN ASSIGNMENTS

1	AA	PROTECTIVE GROUND
2	BA	TRANSMIT DATA
3	BB	RECEIVE DATA
4	CA	REQUEST TO SEND
5	CB	CLEAR TO SEND
6	CC	DATA SET READY
7	AB	SIGNAL GROUND
8	CF	CARRIER DETECT
9	+P	+V. FOR TESTING
10	-P	-V. FOR TESTING
11	CY	ORIGINATE MODE
12	CX	LOCAL MODE
13-19		UNASSIGNED
20	CD	DATA TERMINAL READY
21		UNASSIGNED
22	CE	RING INDICATOR
23-24		UNASSIGNED
25	CN	TERMINAL BUSY